Journal of Pure and Applied Algebra 4 (1974) 353–356. © North-Holland Publishing Company

A CONVERSION ALGORITHM FOR LOGARITHMS ON GF(2^n)

Neal ZIERLER

Institute for Defense Analyses, von Neumann Hall, Princeton, N.J. 08540, U.S.A.

Communicated by J. Stallings
Received 17 December 1973

Abstract. We present a method for expressing a root of one irreducible polynomial of degree n over GF(2) in terms of a basis of GF(2^n) over GF(2) associated with another. This allows us, when both polynomials are primitive, to find logarithms relative to one polynomial from logarithms relative to the other.

Let $f(x)$ be an irreducible polynomial of degree n over GF(2), and let a be a root of f in K = GF(2^n). Then every element of K has a unique expression of the form $h(a)$, where h is a polynomial over GF(2) of degree less than n. If the roots of f are primitive in the multiplicative group of K, then f is said to be *primitive*, and in this case every nonzero element of K also has a unique expression of the form a^r, where $0 \le r \le 2^n - 2$. The number r is said to be the *logarithm* or *exponent* of a^r to the base a or relative to f. Of course, the roots of f are $a, a^2, a^{2^2}, \ldots, a^{2^{n-1}}$, so the logarithms with one root of f as base differ from the logarithms with another as base merely by multiplication by the appropriate power of 2 modulo $2^n - 1$.

The following is the motivating situation. Suppose we have a method for finding the logarithm to the base a of any nonzero member of K given in the form $h(a)$. How can we transpose this into a method for finding logarithms in K relative to some root of a second primitive polynomial $g(x)$ of degree n over GF(2); that is, how can we compute the number t such that $b^t = h(b), 0 \le t \le 2^n - 2$, for some fixed root b of g and for all nonzero polynomials h of degree less than n over GF(2)? Clearly the problem will be solved if we can express b in the a basis; that is, if we can find coefficients $c_0, c_1, \ldots, c_{n-1}$ in GF(2) such that $b = c_0 + c_1 a + \ldots + c_{n-1} a^{n-1}$. Indeed, once we have such an expression, we can use our logarithm finder for f to find s such that $a^s = b$, and we can also express $b^2, \ldots, b^{n-1}$ in the a basis. Then we can convert a polynomial in b into one in a, find its a-logarithm r and thereby its b-logarithm $s^{-1}r$, where s^{-1} denotes the multiplicative inverse of s modulo $2^n - 1$, since $a^r = a^{s(s^{-1}r)} = b^{s^{-1}r}$ (see [1]).

Clearly, finding a root of $g(x)$ is equivalent to factoring $g(x)$ over K, and the basic idea for the method presented here is derived from a procedure of Berlekamp [2, 3] for factoring polynomials over finite fields. Incidentally, the ideas extend

readily to the case where f and g are polynomials over an arbitrary finite field, but we limit the discussion here to the GF(2) case for the sake of the simplicity in the statement and operation of the algorithm that results, and because of the importance of this case in applications.

Let

$$\varphi_j(x) = \sum_{k=0}^{n-1} (xa^j)^{2^k}, \qquad j = 0, 1, \ldots,$$

and let $\bar{\varphi}_j(x)$ be the residue of $\varphi_j(x)$ modulo $g(x)$.

Lemma 1. *Each $\varphi_j(x)$, regarded as a function on K, is a linear function with values in* GF(2).

Proof. If u is in K, then $\varphi_j(u) = \text{trace } ua^j$, and the result follows from well-known properties of the trace. □

Lemma 2. *If u is in K, then $\varphi_j(u) = 0$ for n consecutive values of j if and only if $u = 0$.*

Proof. The result is obvious if $u = 0$. If $u \neq 0$, the n elements $ua^j, ua^{j+1}, \ldots, ua^{j+n-1}$ of K are linearly independent over GF(2). Indeed, if

$$0 = \sum_{i=0}^{n-1} c_i ua^{j+i} = ua^j \sum_{i=0}^{n-1} c_i a^i ,$$

then $0 = \Sigma c_i a^i$, so $c_0 + c_1 x + \ldots + c_{n-1} x^{n-1}$ is divisible by the minimum polynomial f of a. Since the degree of f is n, we must have $c_0 = \ldots = c_{n-1} = 0$. Hence if v is any element of K, there are elements $v_0, \ldots, v_{n-1}$ of GF(2) such that

$$v = \sum_{i=0}^{n-1} v_i ua^{j+i} .$$

Then

$$\text{trace } v = \sum v_i \text{ trace } ua^{j+i} = \sum v_i \varphi_{j+i}(u) ,$$

so the trace vanishes identically on K if $\varphi_{j+i}(u) = 0$ for $i = 0, \ldots, n-1$. But every element v of K such that trace $v = 0$ is a root of $x + x^2 + x^{2^2} + \ldots + x^{2^{n-1}} = 0$, so the trace vanishes for at most 2^{n-1} out of the 2^n elements of K, and the proof of the lemma is complete. □

Lemma 3. *If b is a root of $g(x)$, then $\bar{\varphi}_j(b) = \varphi_j(b)$.*

Proof. For some polynomial $h(x)$,

$$\bar{\varphi}_j(x) = \varphi_j(x) + h(x)g(x),$$

so

$$\bar{\varphi}_j(b) = \varphi_j(b) + h(b)g(b) = \varphi_j(b) + h(b)\,0 = \varphi_j(b). \square$$

From Lemmas 1 and 3 we have:

Lemma 4. *$\bar{\varphi}_j(x)$ takes on values in* GF(2) *at the roots of $g(x)$.* $\square$

Lemma 5. *If b and b' are distinct roots of g, there exists $j < n$ such that $\bar{\varphi}_j(b) \neq \bar{\varphi}_j(b')$.*

Proof. Suppose that $\bar{\varphi}_j(b) = \bar{\varphi}_j(b')$ for $j = 0, \ldots, n-1$. By Lemma 3, the same is true with the bars removed, and by the linearity of φ_j this is the same as $\varphi_j(b-b') = 0$ for $j = 0, \ldots, n-1$, and it follows now from Lemma 2 that $b = b'$. $\square$

The algorithm

Step 0: Set $g_0(x) = g(x)$.

Step j, $j = 1, 2, \ldots$: Compute the polynomial $h_j(x)$ in $K[x]$ defined by

$$h_j(x) = \gcd(g_{j-1}(x), \bar{\varphi}_{j-1}(x)).$$

If $h_j(x)$ is of first degree, say $h_j(x) = ux + v$, then $b = v/u$ is a root of g expressed in the desired form, since arithmetic in K is done in the a basis, and we are finished. If $h_j(x)$ is of degree 0, set $g_j(x) = g_{j-1}(x)$, and go on to step $j+1$. Otherwise – that is, if degree $h_j(x)$ is greater than 1 – set $g_j(x) = h_j(x)$ and go on to step $j+1$.

Maximum number of steps: $j = n$.

In order to prove that the algorithm works, we need:

Lemma 6. *Suppose that we have gotten through Step j without finishing, so that $g_j(x)$ has been defined and is of degree greater than 1. Then for any two roots b and b' of $g_j(x)$, $\bar{\varphi}_k(b) = \bar{\varphi}_k(b')$ for $k = 0, \ldots, j-1$.*

Proof. The statement is vacuously true for $j = 0$. Suppose $j > 0$ and the assertion is true for $j-1$. Observe first that $g_j(x)$ is a divisor of $g_{j-1}(x)$, so if b and b' are two roots of $g_j(x)$, it follows from our inductive assumption that $\bar{\varphi}_k(b) = \bar{\varphi}_k(b')$ for $k = 0, \ldots, j-2$. Thus we need only prove that $\bar{\varphi}_{j-1}(b) = \bar{\varphi}_{j-1}(b')$. If $h_j(x)$ is constant, $\bar{\varphi}_{j-1}(x)$ takes the value 1 at both the roots b and b' of g (and g_{j-1}), so the assertion is true for $g_j(x) = g_{j-1}(x)$. If $h_j(x)$ is of degree greater than 1, $\bar{\varphi}_{j-1}$ takes the value 0

at all of its roots, so the assertion is true for $g_j(x) = h_j(x)$. This completes the proof of the lemma. □

It follows at once from Lemma 6 that the algorithm must yield the desired result for $j < n$, since otherwise $g_n(x)$ would be defined as a divisor of $g(x)$ of degree greater than 1 such that if b and b' are two of its roots, $\bar{\varphi}_k(b) = \bar{\varphi}_k(b')$ for $k = 0, \ldots, n-1$, and this is impossible for $b \neq b'$ by Lemma 5.

References

[1] T.C. Bartee and D.I. Schneider, Computation with finite fields, Information and Control 6 (1963) 79–98.

[2] E.R. Berlekamp, Factoring polynomials over finite fields, Bell System Tech. J. 46 (1967) 1855–1859.

[3] E.R. Berlekamp, Factoring polynomials over large finite fields, Math. Comp. 24 (111) (1970) 713–735.